CECS 2200-20-p1c1

Web: En Blackboard, o directamente lmvicente.com/cecs2200/cecs2200.htm

CHAPTER 1 Introduction to Computers and Programming 1

CHAPTER 1 Introduction to Computers and Programming 1

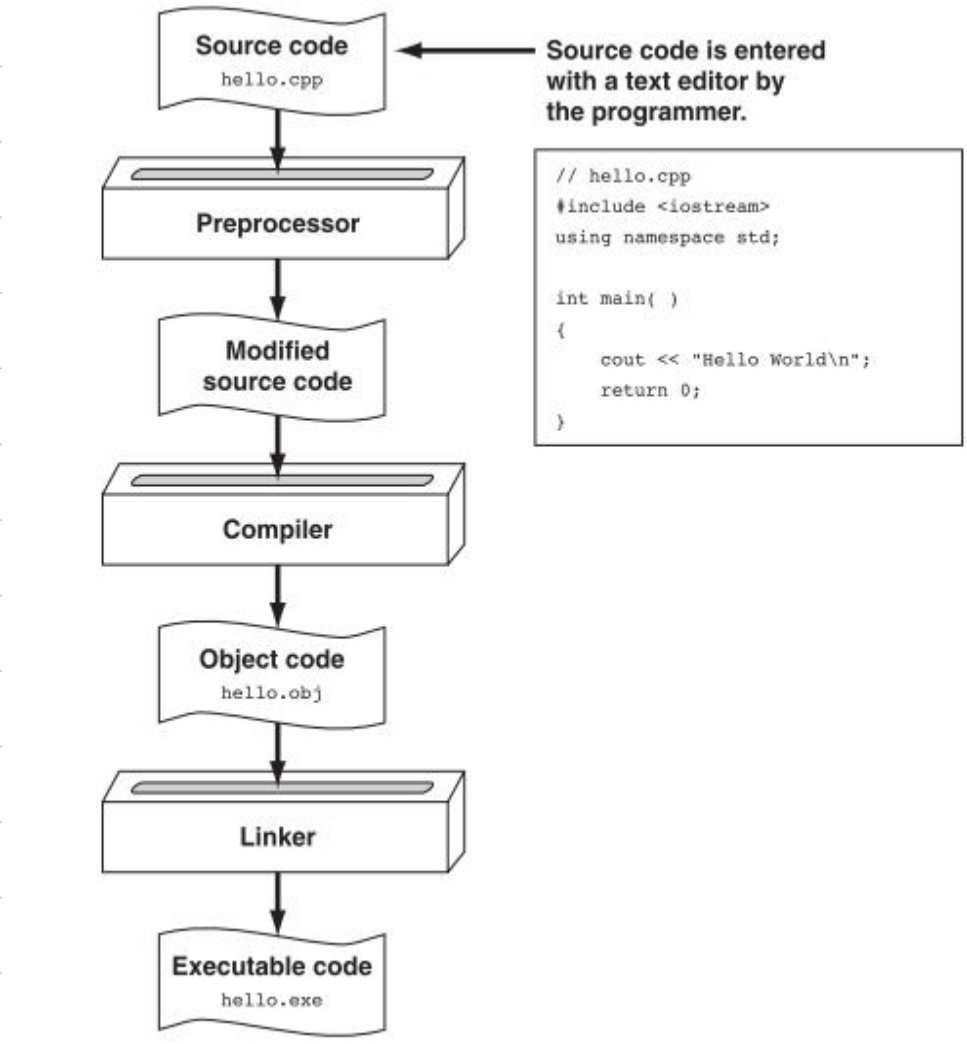
1.1 Why Program? 1
1.2 Computer Systems: Hardware and Software 2
1.3 Programs and Programming Languages 6
1.4 What Is a Program Made of? 12
1.5 Input, Processing, and Output 16
1.6 The Programming Process 17
1.7 Tying It All Together: *Hi! It's Me* 22

Source Code, Object Code, and Executable Code

When a C++ program is written, it must be typed into the computer and saved to a file. A *text editor*, which is similar to a word processing program, is used for this task. The statements written by the programmer are called *source code*, and the file they are saved in is called the *source file*.

After the source code is saved to a file, the process of translating it to machine language can begin. During the first phase of this process, a program called the *preprocessor* reads the source code. The preprocessor searches for special lines that begin with the # symbol. These lines contain commands, or *directives*, that cause the preprocessor to amend or process the source code in some way. During the next phase the *compiler* steps through the preprocessed source code, translating each source code instruction into the appropriate machine language instruction. This process will uncover any *syntax errors* that may be in the program. Syntax errors are illegal uses of key words, operators, punctuation, and other language elements. If the program is free of syntax errors, the compiler stores the translated machine language instructions, which are called *object code*, in an *object file*.

Although an object file contains machine language instructions, it is not a complete program. Here is why. C++ is conveniently equipped with a library of prewritten code for performing common operations or sometimes-difficult tasks. For example, the library contains hardware-specific code for displaying messages on the screen and reading input from the keyboard. It also provides routines for mathematical functions, such as calculating the square root of a number. This collection of code, called the *run-time library*, is extensive. Programs almost always use some part of it. When the compiler generates an object file, however, it does not include machine code for any run-time library routines the programmer might have used. During the last phase of the translation process, another program called the *linker* combines the object file with the necessary library routines. Once the linker has finished with this step, an *executable file* is created. The executable file contains machine language instructions, or *executable code*, and is ready to run on the computer.



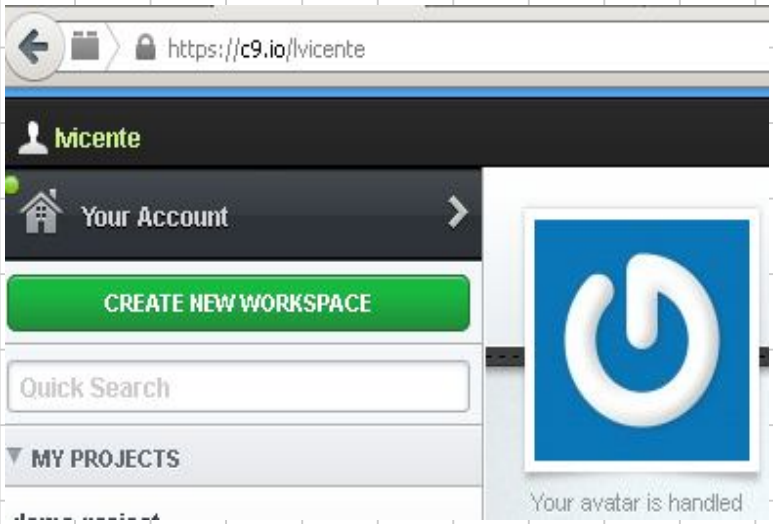
1.4 What Is a Program Made of?

CONCEPT: There are certain elements that are common to all programming languages.

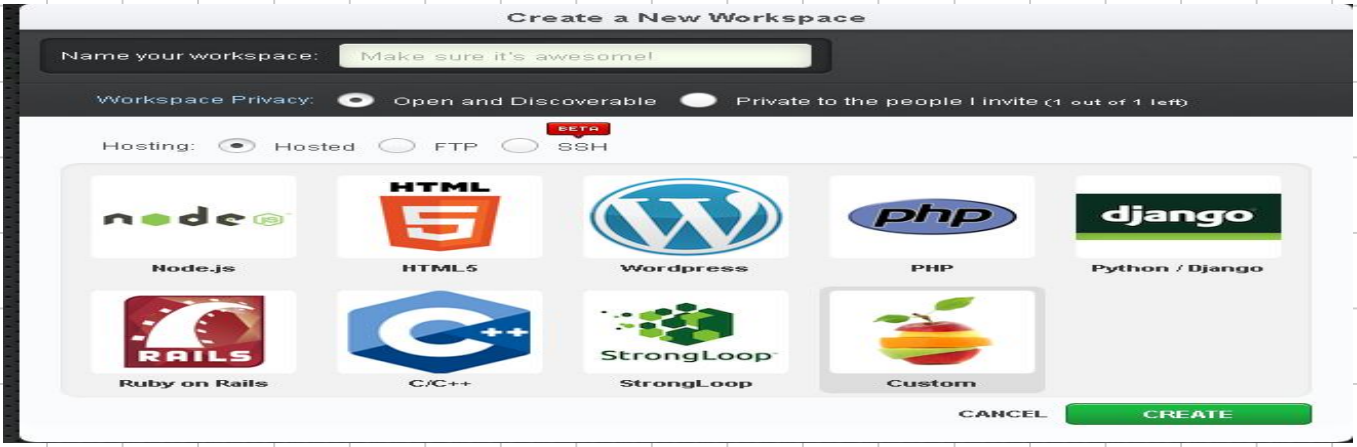
Table 1-2 Programming Language Elements

Language Element	Description
Key Words	Words that have a special meaning. Key words may only be used for their intended purpose. Key words are also known as reserved words.
Programmer-Defined Identifiers	Words or names defined by the programmer. They are symbolic names that refer to variables or programming routines.
Operators	Operators perform operations on one or more operands. An operand is usually a piece of data, like a number.
Punctuation	Punctuation characters that mark the beginning or ending of a statement, or separate items in a list.
Syntax	Rules that must be followed when constructing a program. Syntax dictates how key words and operators may be used, and where punctuation symbols must appear.

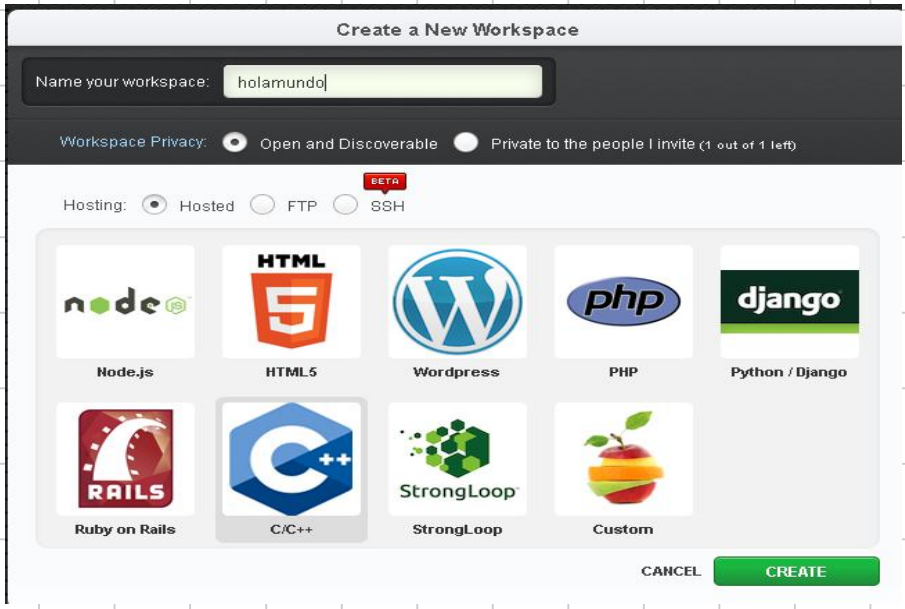
Vamos a usar un compilador en la web: https://c9.io
Háganse una cuenta y les va a salir esto:

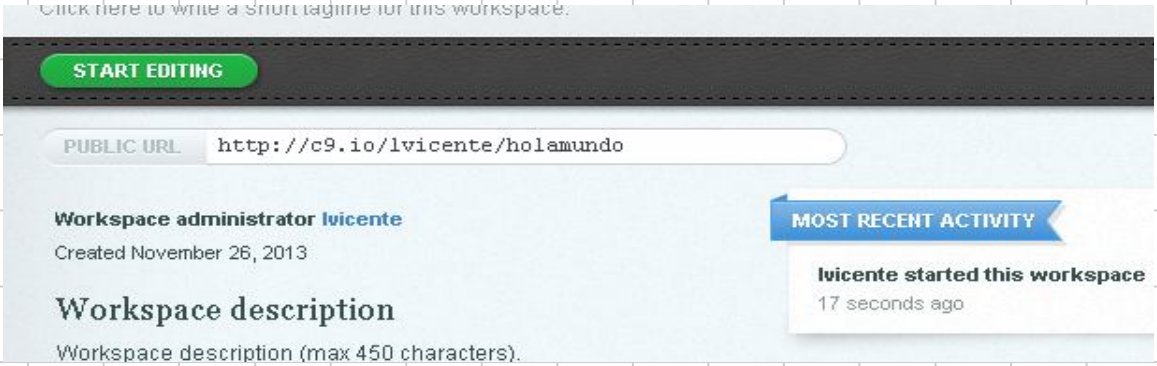


Pinchar en CREATE NEW WORKSPACE

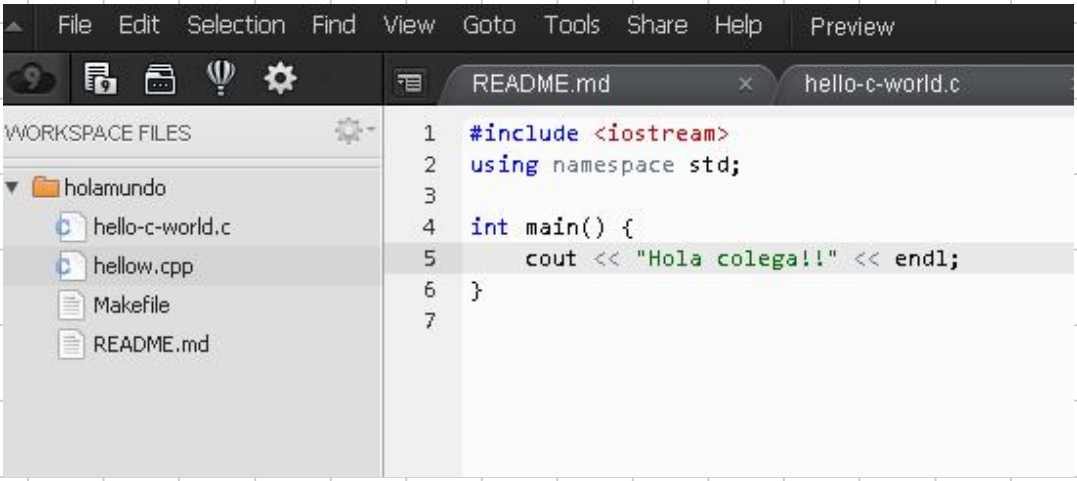


Pinchen en C++, le dan un nombre y pinchen en CREATE





Pinchar en START EDITING



Una vez hecho el codigo fuente "source code", hay que compilarlo y linkearlo

en la ventana de comando escribir:
g++ hellow.cpp -o hellow.exe

Una vez compilado se invoca al ejecutable con:
./hellow.exe